An Image Processing Tool for Cropping and Enhancing Images

by

Ghaleb Abdulla

Report submitted to the faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

© Ghaleb Abdulla and VPI & SU 1993

APPROVED:

Edward A. Fox

Prof. Edward A. Fox, Chairman

Clifford A. Shaffer

Laurence W. Carstensen

December, 1993

Blacksburg, Virginia
An Image Processing Tool for Cropping and Enhancing Images

by

Ghaleb Abdulla

Committee Chairman: Prof. Edward A. Fox

Department of Computer Science

(ABSTRACT)

An educational system called GeoSim is being developed at Virginia Tech; its purpose is to simulate processes related to several geographical subjects. The software consists of six different modules; one of these modules is designed to simulate a field trip for orienteering and position finding. This module uses a database of captured images to pan and zoom from one location to another. However, the original images have overlapping areas which does not allow simulating a continuous panoramic view. To fix this problem a cropping tool was designed and implemented with Intel DVI ActionMedia boards to support the orienteering module of project GeoSim. The tool allows cropping of overlapped areas in the images. In addition, the tool allows the user to minimize differences in intensity and colors between neighboring images. The cropping, color, and intensity values obtained from manipulating images are saved in an ASCII file where they can be read and used in the orienteering module. The images used are captured and stored on the hard disk from a videodisc, with 512x480 resolution and 16 bits per pixel DVI compressed format.
ACKNOWLEDGEMENTS

I wish to express my sincere thanks to Dr. Fox, my advisor and committee chairman, for his continuous support and assistance. Without his patience and cooperation throughout this research it would not have been possible to finish this project. I would also like to thank Dr. Laurence W. Carstensen for serving on my committee and for his help.

My special thanks go to Dr. Clifford A. Shaffer for his support, comments, and cooperation. His sincere efforts and comments during the implementation of this project were a major help.

Special thanks to Intel Corporation for their generous donation of the DVI hardware and software and for their support.

I also would like to thank Praveen Paripati for his helpful discussions and comments.

I wish to thank my wife Reema Nayfeh for her unlimited support during this difficult period. I also wish to thank my children Amer and Sarah for their joyfulness and brightness.

Finally special thanks go to my parents Mohammad and Sarah Abdulla for their support, guidance, and encouragement.
# TABLE OF CONTENTS

1 INTRODUCTION
   1.1 GeoSim .................................................. 1
   1.2 Orienteering ........................................... 1
   1.3 Scope of the Report .................................... 3
   1.4 Outline of the Report ................................... 3

2 BACKGROUND
   2.1 Interactive Multimedia Applications ...................... 4
   2.2 DVI Technology .......................................... 5
      2.2.1 Hardware Overview .................................. 5
      2.2.2 Software Overview .................................. 6
      2.2.3 Graphics Concepts .................................. 6
   2.3 Development Environment ................................ 10

3 PREVIOUS WORK
   3.1 The Videodisc Prototype .................................. 11
   3.2 The DVI Prototype ....................................... 13
   3.3 System and Software Update ................................ 13

4 CROPPING TOOL
   4.1 Introduction ............................................. 15
   4.2 User Interface ............................................ 17
      4.2.1 Design Details ....................................... 18
   4.3 Implementation Details .................................. 26
CONTENTS

4.3.1 Memory Limitations ........................................... 26
4.3.2 Image Processing Techniques ................................. 31
4.3.3 Output Data File .............................................. 33
4.4 Image Enhancements Techniques ............................... 34

5 CONCLUSION AND FUTURE WORK ......................... 38

A Program Flowcharts ........................................... 40

B Data Flow Diagram ............................................ 43

C Source Code Modules ........................................... 45
LIST OF FIGURES

2.1 A DVI bit map, adapted from software library overview [9]. 7

4.1 The shaded areas in image A and image B are redundant. 16
4.2 The two images after cutting the redundant area and pasting them together. 16
4.3 The images numbered according to viewing angles. 19
4.4 The main screen; the twelve images are not yet cropped. 20
4.5 The main screen; the twelve images after they have been cropped. 21
4.6 Images after a circular left shift. 23
4.7 Images after circular right shift. 24
4.8 Image for view at angle 30 before processing. 27
4.9 Image for view at angle 30 after processing. 28
4.10 Initial map for the DVI VRAM. 29
4.11 DVI VRAM map, after allocating space for the bit maps. 30
4.12 Sample of the output data file. 34

A.1 Flow chart for the First Screen. 41
A.2 Flow chart for the Second Screen. 42

B.1 The main components of the tool. 44
LIST OF TABLES

C.1 The source code modules for the cropping tool. .......................... 46
Chapter 1

INTRODUCTION

This paper describes the design and implementation of an automatic cropping tool for manipulating captured images from a videodisc to eliminate redundancy and to minimize color and intensity differences between images. The tool was designed as part of the GeoSim project and implemented on a 386 PC equipped with the Intel Action Media boards using Microsoft C/C++ version 7.0.

1.1 GeoSim

GeoSim is a multi-disciplinary effort by several departments at Virginia Tech to teach geography with the aid of computers. The project focuses on several computer lab exercises which are designed to teach students the concepts of dynamic spatial processes embodied in five fundamental themes of geography. The project combines the information presentation and analysis capabilities of geographic information systems (GIS) with the interactive techniques of computer simulation [4]. One of the developed modules is a computer exercise that simulates orienteering.

1.2 Orienteering

In orienteering, students are given a map and a compass, and must determine their position in an unknown area and navigate through the area until they reach a certain goal [13]. The students should be able to identify their position within this area and find the most efficient route to the goal. Traditionally, teaching orienteering in a geography class involves trips by the instructor to the area where the students will do the orienteering exercise. The
CHAPTER 1. INTRODUCTION

instructor sets up locations and marks for the students to follow. The students go to the area and perform the exercise. This process consumes a great deal of time and effort. Weather conditions may be a major obstacle to performing the exercise. Handicapped students may not have the ability to perform such exercises. These disadvantages were among the motives for developing the orienteering module as part of GeoSim. Although computer simulation does not provide all the challenges that might be part of such trips, still it supports learning of the basic ideas behind such an exercise, and give students practice with using maps.

The original version of the orienteering module was developed and used at Virginia Tech using videodisc technology. Originally, videodisc systems were developed to compete with video cassette recorders, although they did not succeed in the competition for several reasons [7]. The education field offered better opportunities. Videodiscs are used extensively as storage devices for educational or training material that can be accessed and controlled by computer. The ability to access data randomly and to control the videodisc by a computer opened a new range of computer applications. Depending on the way the user controls the videodisc, the current range of videodisc systems can be divided into five levels [7]. In level 3, the level used in a videodisc version of the orienteering module, the videodisc can be controlled by the user as a normal videodisc player where the images can be viewed sequentially in regular, double, or half speed. Alternatively, it can be controlled by an interactive computer program developed especially for this purpose. In the latter method, viewing the disc contents can be done randomly as the user chooses. This architecture is widely used in education and training programs. The orienteering module is a good example of using the videodisc in education.

A new implementation of this module was developed at Virginia Tech based on the Digital Video Interactive (DVI) Action Media boards. The new implementation uses the same set of images used in the videodisc prototype implementation [14], but uses DVI hardware to display images from a hard disk.
CHAPTER 1. INTRODUCTION

1.3 Scope of the Report

The objective of this report is to describe the work done to design and implement a cropping tool that can be used to manipulate captured images using DVI hardware and software. The output of the tool will be used with the orienteering tool which was developed by another student at Virginia Tech [14]. The report also gives a brief technical discussion of the DVI graphics system. The aim of this discussion is to help the reader understand the implementation and to serve as additional documentation for the DVI system.

The user interface in the cropping tool was designed to be simple, flexible, and easy to use. The user can manipulate the images with a minimum number of keystrokes. The tool provides the user with a way to reverse actions. The screen layout is simple and easy to follow. The user can easily load any set of images which corresponds to any location in the orienteering module and can choose any of these images for manipulation. The results of image manipulation are saved in a separate file that can be accessed whenever the user invokes the tool. The image values can be read into a structure where they can be used before subsequent display of processed images.

1.4 Outline of the Report

The rest of the report is organized as follows. The second chapter gives background information about DVI technology, including DVI software, DVI hardware, and DVI graphics concepts. The second section gives a description of the development environment. Chapter Three describes the previous work done on projects related to this project. Chapter Four gives a detailed description of the cropping tool and its design and implementation. That chapter ends with a discussion of image enhancement techniques. Chapter Five presents the conclusions of the report and discusses future work that can be done in this field.
Chapter 2

BACKGROUND

This chapter gives background technical information on the following topics:

- Interactive Multimedia Applications
- DVI Technology
  - Hardware
  - Software
  - Graphics Concepts
- Development Environment

2.1 Interactive Multimedia Applications

Developing interactive multimedia applications is becoming an important area in the software industry because it opens new fields to using computers. A major problem with multimedia applications is the amount of space required to store the media objects. For example, the orienteering module has 4620 images [3], each image requires approximately 500 Kbytes of disk space, consequently, we need 2,310 Megabytes. Because computers are getting faster and cheaper, it is practical to compress the data before storing it. Using the DVI compression techniques, the size of each image will be 100 Kbytes, requiring 462 Megabytes for 4620 images instead of 2,310 Megabytes for the uncompressed data. In this case, using compression saves lots of space. Also reading 100 Kbytes from the disk takes one-fifth the time needed to read the uncompressed image. The decompression is done by
CHAPTER 2. BACKGROUND

the dedicated DVI hardware so quickly that reading and decompressing takes less time than reading an uncompressed image.

2.2 DVI Technology

DVI technology is a combination of hardware and software that allows creating multimedia applications. In October 1988 Intel bought DVI technology from GE. Since then Intel has committed itself to making this new technology standard for PC's. Different projects used DVI boards to implement software applications. Typical applications that can be developed using DVI technology are surrogate travel and synthetic video [12]. With surrogate travel users can sit in the lab and use the computer to visit distant sites. The difference between interactive and passive use is that with computers the user can control the watching sequence. GeoSim is an example of this type of application. A computer equipped with an audio/video system is capable of creating synthetic video images by combining a computer three-dimensional model with real video images of surfaces, patterns, and textures [12].

2.2.1 Hardware Overview

Action Media hardware consists of the delivery board and the capture module. In the latest DVI hardware and software release, the two boards are mounted together to form one board that will occupy one PC slot. The main function of the two boards is to convert analog-source audio and video elements into compressed digital data and to enable application playback. Storing the data in digital format gives great flexibility to manipulate it. For example, the data can be edited, modified, cut, and pasted. The heart of the DVI boards is a chip set which consists of two chips, the display processor and the pixel processor. The display processor is used for the display functions and data extraction from the Video Random Access Memory (VRAM). The pixel processor is used for real time data compression/decompression.
CHAPTER 2. BACKGROUND

2.2.2 Software Overview

The Action Media II system software contains the following:

- DOS media preparation utilities which may be used to capture still video, and compress and edit the audio/video.

- Software libraries which support system calls that may be used from the C language or by authoring software to communicate with the system.

- Action Media device drivers which control the VRAM, SCSI bus, audio, and video capabilities of the delivery board.

2.2.3 Graphics Concepts

This section draws heavily on the ActionMedia software library overview manual [9].

Digital video graphics images are represented in the VRAM by a collection of pixel values called a bit map. Figure 2.1 represents the format of a bit map [9]. Each pixel value in the bit map defines the color and intensity of the corresponding pixel in the video display. The resolution of the bit map is expressed as the number of pixel values in the horizontal and vertical dimensions. The pixel values in a bit map are arranged sequentially. The first pixel value in the bit map corresponds to the upper left pixel on the video display. This pixel value is important because it corresponds to the bit map start address; the other pixel values follow sequentially from left to right and top to bottom. The distance between a pixel address in one line of the bit map and the corresponding pixel address in the next line is called pitch [9], usually expressed in units of pixels. A bit map is defined so that the X resolution must be less than or equal to the pitch. If the X resolution of the bit map is equal to the pitch then the image will fill every pixel in the memory continuously, while if it is less than the pitch then the bit map memory will have empty spaces. If the bit map is displayed on the screen, then the empty spaces in memory represent the empty spaces on the screen between the displayed image and the edges of the screen. The memory used in the bit map
CHAPTER 2. BACKGROUND

Figure 2.1: A DVI bit map, adapted from software library overview [9].
CHAPTER 2. BACKGROUND

can be imagined as a two dimensional space with the displayed image smaller than the bit map, represented as a smaller square or rectangle inside the big rectangle (the bit map). This seems a waste of space, but with planned use of the memory a programmer can fit one or more images in these gaps. The bit map can be fully represented by a resolution and a number expressing the bits/pixel. This number is called the pixel format and it defines the visual properties of the image such as color and intensity.

There are different types of bit maps such as those referring to color look-up tables, packed bit maps, and planar bit maps. In the cropping tool a 16 bit packed bit map is used, which is the simplest but least efficient representation supported since it consumes lots of memory. This format is suitable for graphics applications and programmed animation and it can produce video quality still images. As was mentioned, this bit map uses the 16 bits/pixel presentation; the 16 bits are divided into 6 bits for Y, 5 bits for U and 5 bits for V, arranged as follows:

UUUUUUVVVVYYYYY

The Y component, which is the least significant in this arrangement, is more important than the other components, and thus six bits are used to represent it.

Color Fundamentals

Color coding is easily understood if the color is visualized as a composite signal of three basic colors: red, green, and blue. This coding scheme is easy to use and can give a quantitative description of the color. A major disadvantage of this coloring scheme is the difficulty of transmitting three different signals in TV transmission. The solution was to come up with a composite signal from these three colors which can be transmitted on a single wire or as a single signal. This composite format is used in studios and in television transmission [12]. In the late fifties when color TV was introduced to replace monochrome TV, the NTSC (National Television Standards Committee) standard was introduced. This standard was designed to make use of the composite signal and the ability to divide this
CHAPTER 2. BACKGROUND

signal into two parts: luminance and chrominance. The luminance part is a monochrome signal that controls the brightness of the signal. The chrominance part contains the color information. The chrominance is further divided into two signals called the color difference. In the NTSC system the luminance signal is called the Y signal, and the two chrominance signals are I and Q. The YUV system is very similar to the NTSC system, with U and V components instead of the I and Q components.

The YUV color space is an international color standard and it is related to the RGB color space by a linear transformation. In this color standard, the Y component contains all the brightness and intensity information of a pixel (luminance). The U and V specify the color of the pixel. The Y component in this standard is always positive, while the U and V components can be positive or negative. The bit map values stored in the DVI system are unsigned integers; as a result, the UV values which are stored are offsets from a U or V value and the stored values will be u+m or v+m where m can be in the range from -n to n. In DVI, for a 24 bits/pixel format, the Y, V, and U will each be represented as 8 bits/pixel and encoded as follows: Y: 16 to 235, where “zero” (which represents no intensity) = 16; U,V: 16 to 240, and “zero” (which represents no color) = 128. So a YUV value of (16,128,128) represents the color black: no intensity and no color.

Using the YUV coding has several advantages. First, the YUV encoding takes advantage of an important property of the human visual system: in high frequency detail the eye is more sensitive to the intensity or brightness of the image than to its colors. Thus the image can be stored taking less space if we give the U,V signals a lower resolution value. Another advantage is that this encoding can be used to do some interesting image processing. For example we can convert a colored image to monochrome by simply storing the value 128 in the U,V components. The Y value can be used to produce interesting effects like reversing the image lighting by XORing the Y value with all 1’s.
CHAPTER 2. BACKGROUND

2.3 Development Environment

The system used for developing the software is an Intel ActionMedia AT compatible unit, which has a 386/25 MHz Intel processor. The operating system is DOS 5.0. The compiler used is Microsoft C/C++ 7.0. The system has 4 Megabytes of RAM and 2 Megabytes of DVI Video RAM, a 560 Megabyte SCSI hard disk, and a second 50 Megabyte hard disk. The system has the Intel ActionMedia II boards and Intel 2.20 DVI software.

The DVI system was chosen for two reasons: first, the availability of the hardware which was donated by Intel Corporation; second the DVI hardware and software give a powerful environment for multimedia applications. The software comes with some support tools like capture utilities, and it comes with C libraries that can be used easily with Microsoft C.
Chapter 3

PREVIOUS WORK

The orientation, position finding, and orienteering module, one of the modules in the GeoSim project, was developed at Virginia Tech. This module has been implemented using two different environments, the videodisc and the DVI hardware.

3.1 The Videodisc Prototype

The videodisc prototype was developed in the late eighties by Carstensen and Cox [3]. The system used for this prototype was an IBM-PC with attached videodisc player. The videodisc is controlled by the computer with the capability of superimposing computer graphics over the video images. The system had an RGB monitor connected to the display adapter, and a composite monitor connected to the videodisc. A computer program controls the interaction with the videodisc. This configuration was used successfully in teaching orienteering and position finding.

Before developing computer simulation software, lots of work has to be done in the design and preparation phases. First, a study area to be used in the simulation was chosen, to satisfy certain requirements [3].

Second, a field map with a desired scale was prepared for this area. The possible movements in the area were studied and a hexagonal grid was drafted at the correct scale and overlaid over the field map. The center of each hexagon in the map was considered as a location and for this location there are at maximum six neighboring locations representing the centers of the surrounding hexagons. Standing at any locations, the student can move forward to any of the six neighboring locations and can turn his head to look around.
CHAPTER 3. PREVIOUS WORK

Third, 385 stakes were laid within the study area, one at the center of each hexagon in the field map. Laying the stakes required a visit to the area by the designers.

Finally, photographs were taken of the panoramic views from each stake placed during the last step. The study area was photographed from a level tripod at each of the 385 locations. For each location, twelve photographs were taken beginning at magnetic north and following every 30 degrees.

The images are accessed from the videodisc player and displayed on the screen using a computer program. Using the left and right arrow keys the student can simulate turning and looking around. For example, if the student uses the left arrow then the image which is located left of the current image will be displayed. The right arrow key displays the image to the right of the current image. The main screen has a compass in the right upper corner which indicates the direction of looking by a needle that points in a certain angle. The needle will be colored red if there is no neighboring cell at this angle or green when there is a neighboring cell. The up arrow key will allow the user to move forward into a neighboring cell whenever the compass arrow is green.

The way the images are stored on the videodisc will affect the performance. For example, if the images are accessed sequentially, in a similar way to how they are stored on the videodisc, the time needed to move the read head to get the next image is tolerable. On the other hand, accessing an image located far from the current image will be slow, since we have to move the reading head from the current location to a location where the target image is stored. An example of the latter case is when simulating turning the head 180 degrees. Using the videodisc to do this simulation means that the target image should not be placed very far from the current image. As a result the software designer has to worry about the location of images on the videodisc. Normally this is done by studying all access paths in such applications and placing the images on the videodisc in a way that will give the fastest access time. With DVI, which can manipulate digitized video and images, the access time is faster since images are stored on the hard disk. This was one of the motives to move to a DVI implementation for the orienteering module.
CHAPTER 3. PREVIOUS WORK

3.2 The DVI Prototype

The module runs on a 386 system which has Intel Action Media boards. As in the videodisc prototype, the software simulates the orienteering and position finding exercise by using images taken at the required locations. The user interface for the DVI prototype is very similar to the videodisc prototype user interface. However, this prototype is implemented using DVI hardware and as a result the speed of this system is better. In addition, using the DVI hardware allowed implementing some transition effects such as panning and zooming. Panning is used when simulating turning the head to the right or to the left and zooming is used when simulating forward movement [14]. An option that can be available by using the DVI system, is the ability to store images on a CD-ROM. Using the CD-ROM should not affect the performance of the system, since the hard disk can be used as a buffer to store images read from the CD-ROM before they are displayed.

3.3 System and Software Update

The DVI prototype for the orienteering module was developed first using DVI software version 2.13. My initial work for this project involved updating the system with the new hardware (i.e., ActionMedia II boards) and software version 2.2, updating the C compiler, recompiling the code for the orienteering module, and enhancing the zooming effect. Because the new hardware is faster than the old, the panning and zooming effect became faster. After the new hardware was installed, the first step was to recompile the module and make sure that it worked with the new boards; this step was done with few problems. The next step was to change the zooming effect to look more natural, which was done through three steps. First, a buffer is used to decompress and load the target image before starting the effect. Second, the target image is displayed by copying it on the screen instead of using the iris effect, since this transition is faster. Third, the number of steps to scale the original image was increased to give the zooming a more natural feel. The scaling process is used to give a transition effect before displaying the new image. Scaling the image with an expand option
CHAPTER 3. PREVIOUS WORK

will produce a zoom effect with blocky pixels because the expand operation inserts averaged in-between pixels. Increasing the number of scaling steps over a certain number will produce an undesirable effect since the averaged pixels will grow bigger. In the orienteering module the number of steps was increased to 6. The blocky pixels effect will not start appearing until the number of steps is greater than 10.
Chapter 4
CROPPING TOOL

4.1 Introduction

For each location in the orienteering module we have 12 images or pictures. When placed beside each other, these images should form a circle. However, since they have overlapped areas and color differences they do not form a continuous seamless circle. Before any image can be displayed using the orienteering module, the amount of overlap between the images must be measured and at display time it should be cropped.

To test the orienteering module, a set of images were cropped using a tool that was developed specially for this purpose. The tool is used to display two neighboring images on the screen, finding a position which is shared between the two images, measuring the distance using a ruler, and then converting the distance from inches to pixels. The cropping tool was developed to allow for processing images to provide a continuous panoramic view from the twelve images in an easier way. The images should be processed without modifying them, i.e. the result of processing an image should be seen on the screen only and it should not change the image stored on the hard disk. This will allow for retrieving the original images at any time in the future and modifying them again. The result of manipulating the images can be expressed in numbers and can be saved in an ASCII data file that can be used by the orienteering module to crop and enhance images before they are displayed. A discussion of the data file and its structure will follow in this chapter.

Figure 4.1 shows two images $A$ and $B$; assume that image $A$ was shot at angle 0 and image $B$ was shot at angle 30. The shaded area in $A$ and the shaded area in $B$ are redundant. Placing the two images beside each other on the screen will not produce a continuous scene.
CHAPTER 4. CROPPING TOOL

Figure 4.1: The shaded areas in image A and image B are redundant.

Figure 4.2: The two images after cutting the redundant area and pasting them together.
CHAPTER 4. CROPPING TOOL

To solve this problem the shaded part from one of the two images, either $A$ or $B$, should be cut and the two images combined. If the images are considered as sets then the process of cutting one of the shaded areas is similar to a union operation between the two sets $A$ and $B$. Figure 4.2 shows the effect of this operation on the two images; the two images now share the area which was redundant.

The cropping tool allows performing such operations on captured images. The images, which are stored in DVI 16 bit compressed format, are processed using the cropping tool to eliminate redundancy in the scenes. The user can load any two images from a set of twelve images, which corresponds to a cell, and crop the left side of the right image until it matches the right side of the left image.

The user interface is easy to use and it allows the user to do the cropping quickly and accurately. The user will view the effect of image cropping on the screen but this will not affect the images saved on the hard disk. The user can also control the brightness, contrast, tint, and saturation of the image. The values for these parameters and the crop value are saved in an ASCII file. The rest of this chapter will discuss all these ideas in detail.

4.2 User Interface

A major goal in the design of a user interface is to make it simple, easy to use, flexible, and to give the user full control while using the interface. In addition it should provide the expected functionality; that is, the user should be able to use the software to achieve the intended goal of the software.

The user interface for the cropping tool was designed to satisfy all these design goals. The tool provides all the expected functions to the user through two screens. The first screen allows the user to choose an image for manipulation and to review the effects of the work. The second screen allows the user to perform the required modifications on the image like cropping and changing the “perception parameters”.

17
CHAPTER 4. CROPPING TOOL

4.2.1 Design Details

Figure 4.3 shows an outline of the Main Screen; the upper part of the screen contains the title message. The middle part contains the twelve images which were taken at angles that are multiples of 30, for example the top left image is the image at angle 0 and the next image is the image taken at angle 30. When the session starts, the user should enter a selected cell number. If the images corresponding to this cell were not modified before then the user will see a screen as shown in Figure 4.4. The black border between images on the same row is actually part of the right image of the pair. If the images were cropped then the screen might look like Figure 4.5. A red square will appear surrounding one of the images (Figure 4.5 shows a black square surrounding the topmost left image); that square can be used to choose an image for cropping. The lower part of the screen contains messages that list the available options for the user.

The user can interact with this screen using the following keys:

- Left arrow key: this key will move the red square backwards one image. If the current image is the topmost left image (image 0 in Figure 4.3) then this key will place the square on the lower right image (image 330 in Figure 4.3). If the current image is the leftmost image on any of the second or third group of images (image 120 or image 240 in Figure 4.3) then this key will move the square to the last image in the previous line (image 90 or image 210 in Figure 4.3).

- Right arrow key: this key will move the red square forward one image. If the current image is the lowermost right image (image 330 in Figure 4.3) then this key will place the square over the left upper image (image 0 in Figure 4.3). If the current image is the rightmost image on the first or second group (image 90 or image 210 in Figure 4.3) then this key will move the square to the first image in the next group (image 120 or image 240 in Figure 4.3).

- The letter “l”: this key will cause a circular shift to the left of all the images on the screen. This helps when an image like the leftmost image on the second row is
### Figure 4.3: The images numbered according to viewing angles.
CHAPTER 4. CROPPING TOOL

![Image of various hillside images with instructions]

Use the right arrow key and hit Return to choose an image
Use the 'r' key to rotate the images right
Use the 'l' key to rotate the images left

Figure 4.4: The main screen; the twelve images are not yet cropped.
Figure 4.5: The main screen; the twelve images after they have been cropped.
CHAPTER 4. CROPPING TOOL

manipulated and we need to see the result by placing it directly beside the image located at the rightmost of the first row. Figure 4.6 shows the effect of this key.

- The letter “r”: this key will cause a right circular shift. The effect of rotating the images right helps in a way similar to the previous key except that the direction of rotation is to the right. Figure 4.7 shows the effect of this key.

- ESC key: this key will quit the program, saving any modifications in the data file.

- Enter key: this key will choose the current image highlighted by the square to be processed and it will activate the second screen, which is used to modify the current image.

Figure A.1 in appendix A shows the flowchart for the first Screen.

Figure 4.8 shows the Second Screen. The image that was chosen for manipulation will appear in the upper right quarter of the screen with a resolution of 256x240. The image to its left will appear in the upper left quarter of the screen. The bottom part of the screen contains the instructions that describe the key actions. When the left image is displayed, the crop value (if not zero) is used to display the image. To crop the right image, the user moves it to the left until the two images make a continuous scene. The user can control the steps that the image can move in terms of pixels. The default value is 1 pixel, and the user can increase the step up to 16 pixels. The user can change the image perception parameters, such as brightness, contrast, tint, and saturation. For this Second Screen, the user interacts with this screen through the following keys.

- The up arrow key: this key will increment \textit{stepsize}, which is the number of pixels to move at each step. The default value is 1 pixel and the maximum value is 16 pixels.

- The down arrow key: this key decrements \textit{stepsize}.

- Left arrow key: this key will move the right image to the left \textit{stepsize} pixels. The user can use this key to match the two images together.
Use the right arrow key and hit Return to choose an image
Use the 'r' key to rotate the images right
Use the 'l' key to rotate the images left

Figure 4.6: Images after a circular left shift.
CHAPTER 4. CROPPING TOOL

Use the right arrow key and hit Return to choose an image
Use the 'r' key to rotate the images right
Use the 'l' key to rotate the images left

Figure 4.7: Images after circular right shift.
CHAPTER 4. CROPPING TOOL

- Right arrow key: this key will move the right image to the right \textit{stepsize} pixels. This key helps when the cropped value is greater than what is necessary; in this case the image can be moved back and the left arrow key can be used again.

- “b” key: activates brightness changes (see “+” and “-” keys).

- “c” key: activates contrast changes (see “+” and “-” keys).

- “t” key: activates tint changes (see “+” and “-” keys).

- “s” key: activates saturation changes (see “+” and “-” keys).

- The “+” key: this key increments the brightness, contrast, tint, or saturation, depending on the letter pressed before using the “+” key. The default parameter is the brightness, so pressing “+” before the “c”, “t” or “s” key will increment the brightness.

- The “-” key: this key decrements the brightness, contrast, tint, or saturation, depending on the letter pressed before using the “-” key. The default parameter is the brightness, so pressing “-” before the “c”, “t” or “s” key will decrement the brightness.

- The letter “n” key: this key loads the next image from the set of the twelve images that corresponds to the cell (location) being processed. The right image, which is the one being manipulated, will be loaded in place of the left image and the next image will be loaded at the upper right quarter of the screen. For example, if we are currently manipulating the image at angle 30, then the screen will show the images at angle 30 on the right side of the screen and the image at angle 0 at the upper left corner of the screen. Pressing this key will cause the image at angle 30 to be loaded at the upper left corner of the screen and the image at angle 60 will be loaded at the upper right corner of the screen. The changes made to the previous image (image 30 in our example) will be saved in the output data file.
CHAPTER 4. CROPPING TOOL

- The Return key: this key will save the modifications done on the image and will go back to the main screen.

Figure A.2 shows the flowchart for the second Screen. Figure 4.8 shows the effect of the cropping process.

4.3 Implementation Details

Three major issues which have affected the implementation are discussed in this section:

- Memory Limitations,
- Image Processing Techniques, and
- Output Data File.

Figure B.1 in Appendix B shows a data flow diagram for the cropping tool.

4.3.1 Memory Limitations

Figure 4.10 shows a map of the DVI VRAM. Of the 2 Megabytes of continuous memory, the first 16K are reserved for the CD-ROM SCSI driver. The next 48K are reserved for the audio channels. The area for audio channels is not shown in the memory map because audio will not be combined with this application, so this area can be used by the application program. The last 128K are reserved for the system and should not be used at all. This leaves 1.8 Megabytes; without care it will not be sufficient for our needs. This project uses 512x480 16 bpp compressed images; the average size of a compressed image is 100K. The size of the uncompressed image is 491,520 bytes. Obviously, three bit maps at most can be defined with this size. In the cropping tool one of the bit maps is defined as a work buffer for loading and decompressing the images. A second bit map is used for the screen display, and the third bit map is used as a buffer to manipulate the images before they are displayed on the screen. Figure 4.11 shows a map of the VRAM used in the cropping tool.
CHAPTER 4. CROPPING TOOL

Figure 4.8: Image for view at angle 30 before processing.
CHAPTER 4. CROPPING TOOL

Figure 4.9: Image for view at angle 30 after processing.
CHAPTER 4. CROPPING TOOL

Figure 4.10: Initial map for the DVI VRAM.
CHAPTER 4. CROPPING TOOL

Figure 4.11: DVI VRAM map, after allocating space for the bit maps.
CHAPTER 4. CROPPING TOOL

The work buffer (WB) bit map is used to load and decompress the image with resolution 512x480. The image is then scaled down to 128x120 or to 256x240 into the bit map Scr2; see Figure 4.11. The 128x120 resolution is used in the first screen when we display the twelve images and the 256x240 resolution is used in the second screen when we process the images. The scaled image is loaded into Scr2 with zero values for the brightness, contrast, saturation, and tint variables. The zero values for the four parameters means that the image should not be changed before displaying it. Displaying the image with the values read from the data file (if different from zero) involves copying the image four times in Scr2 and then copying it a fifth time to Scr1 which is the display screen bit map. The reason for the previous copying steps is that there is no one command in the DVI library that will copy the image and change the four parameters at the same time. The next section will highlight this idea and will describe how this problem was solved.

The memory requirements of the two major screens differ depending on the screen layout. The first screen, where the screen contains twelve images in three rows of four images each, uses an auxiliary buffer to manipulate the images. Rotating the images right or left is one of the cases in which the auxiliary buffer is used to store some images before shifting the rest. For example in the case of shifting left, the topmost left image should be saved in a buffer before we move the rest of the images and then it should be restored and displayed as the bottommost right image. The second case in which the other buffer is used is when the red square is moved to highlight the image that will be chosen. In this case each image under the red square will be copied into the buffer. For example, when the right arrow key is pressed to move the square, the original image will be copied to its original place to delete the square currently surrounding the image. The next image is read into the buffer and the square is displayed over it on the screen.

4.3.2 Image Processing Techniques

In DVI systems, controlling brightness, contrast, saturation, and tint is done through four calls provided by the DVI software library. These calls are:
CHAPTER 4. CROPPING TOOL

- **GrBrightness(Dst,Xd,Yd,Xlen,Ylen,Src,Xs,Ys,Delta):** Here Delta is a signed number; the value of this number will be added to the current Brightness value. This call affects the Y component of the image.

- **GrContrast(Dst,Xd,Yd,Xlen,Ylen,Src,Xs,Ys,Delta):** Here Delta is a signed number; the value of this number will be added to the current Contrast value. This call also affects the Y component of the image.

- **GrTint(Dst,Xd,Yd,Xlen,Ylen,Src,Xs,Ys,Delta):** Here Delta is a signed number; the value of this number will be added to the current Tint value. This call affects the U and V components of the image. The effect of this call is similar to the tint button on the TV.

- **GrSaturation(Dst,Xd,Yd,Xlen,Ylen,Src,Xs,Ys,Delta):** Here Delta is a signed number; the value of this number will be added to the current Saturation value. This call affects the U and V components of the image. This call increments or decrements the amount of colors in the image depending on the value of Delta.

The Dst is the destination bit map handle. Xd,Yd, Xlen, Ylen are the numbers which represent the destination rectangle. The Src is the source bit map; and the Xs,Ys are the upper-left source bit map coordinates.

Although each of these calls gives an easy way to change the image characteristics, it still has a major disadvantage. The process of changing the value of any of these parameters is irreversible. For example, increasing the brightness of an image with value equal to 5 and then decreasing the brightness of the image with value equal to 5 will not return the image back to its previous state. To overcome this disadvantage the original image is saved in a buffer and four variables are defined, one for each parameter. These variables will have a cumulative value equal to the number of steps the parameters have been changed multiplied by the stepsize. For example, the brightness, contrast, saturation, and tint have the following variables: brightness-var, contrast-var, saturation-var, and tint-var. When an
CHAPTER 4. CROPPING TOOL

image is chosen for manipulation for the first time, all these variables for this image will have zero values. Assuming that the increment/decrement stepsize=5, then when the “4” key is pressed for the first time, the following series of actions will take place. First, the brightness-var is incremented by 5, second the value of brightness-var is assigned to Delta, finally the image is copied to a buffer with the function call GrBrightness with the new Delta value. The copy operation should be done once for each other variable, i.e, contrast-var, saturation-var, and tint-var. The reason for this is that the other variables might have values which are different from zero and since we are using the original image then we should always update it with the values set in these variables. The four copy operations are done using a buffer in the VRAM; the image is then copied to the output screen using the GrCopy function call. The GrCopy has the following syntax :

GrCopy(Dst,Xd,Yd,Xlen,Ylen,Src,Xs,YS)

which is similar to the other calls except it does not have a Delta value. If the “key” is pressed again then the brightness-var will be incremented and the previous process is repeated again.

4.3.3 Output Data File

An ASCII file stores the result of manipulating the images on the screen. The file is arranged into 385 sets of numbers which is the total number of locations. Each set contains 60 fixed size numbers; 12 images, for each location, where each image has 5 numbers to express cropping, brightness, contrast, saturation, and tint. The 5 numbers in the first set describe the image taken at angle 0; the second set of 5 numbers describes the image at angle 30; and so on. Since the numbers have a fixed size, accessing them is easy. To access the set of numbers that corresponds to location number 20, we multiply the cell number by the size of the set and add it to the address where the data starts in the file. The data for the cell is read into a data structure in memory. Initially the file contains zeroes, and after the user modifies the images it will contain values that reflects the changes made to the
CHAPTER 4. CROPPING TOOL

Values for image 0

\begin{array}{cccc}
+096 & -010 & +000 & +000 -005 \\
\end{array}

Values for image 30

\begin{array}{cccc}
+058 & +010 & +025 & +000 -005 \\
\end{array}

Figure 4.12: Sample of the output data file.

images. The user may retrieve the original images and manipulate them again by accessing the data file, using any usual text editor, changing all the numbers in the file to zeroes and invoking the cropping tool again.

A sample of the data file is shown in Figure 4.12. As can be seen in the figure, for each image there are five signed numbers stored. The first number is the value to be cropped from the left side of the image. The numbers in the second to the fifth positions represent the brightness, contrast, saturation and tint.

4.4 Image Enhancements Techniques

The previous section discussed how to combine the images together to form one continuous image. The cropping process is not difficult and gives excellent results. However, the process of matching brightness, contrast, saturation, and tint in the two images is not as successful as the cropping process because the image parameters are changed for the image as a whole. When contrast is decreased for example it is decreased for all parts of the image.
CHAPTER 4. CROPPING TOOL

Some of the images can be adjusted so that the border line is indistinguishable and some cannot. An image with sky and ground combined with another image with sky and ground can be controlled to match the ground easily. However, the sky portion of the two images might look different; the opposite is also true, where one can match the sky but cannot match the ground. The problem is that the relative intensity and color can be different for sky than for ground. This can be caused by lighting conditions such as clouds, the location of the sun when the picture was taken, etc. This problem might be solved using low level image processing.

Z. Hussain [8] defines computer vision as "the process of extracting, characterizing, and interpreting information from real world images." He divides the process into several steps:

- Sensing,
- Preprocessing,
- Segmentation,
- Description,
- Recognition, and
- Interpretation.

The second item, "preprocessing", includes the topics of image enhancement and noise suppression [8]. These image enhancement techniques may be used to reduce the differences between combined images in the cropping program.

First, the edge between two consecutive images can be used to identify the type of processing needed and where the processing should start. The edge between the two images can be detected by examining the rate of change of intensity near a pixel; sharp changes in the intensity might indicate an edge [2]. Edge detection might work well in the cropping tool because the images are sufficiently homogeneous that the edge between two consecutive images can be determined on the basis of intensity level discontinuities [6]. Once the edge is
CHAPTER 4. CROPPING TOOL

determined, the next step is to smooth the transition around the edge. One simple technique is histogram equalization. A histogram of an image is used to display the distribution of gray levels in the image. The histogram should show a sudden change in magnitude (either positive or negative) whenever there is a sharp difference between the two images. However, the same magnitude change occurs with any two very distinct objects in the same image, for example a dark green tree next to a bright sky. In such cases even histogram equalization will not be sufficiently powerful to do the job and these algorithms need the support of a huge knowledge based system.

Assuming that the tools are available to detect the edge between two images, manually or by a knowledge based system, then one of the following techniques can be used to minimize the differences between the two images [6]:

- Neighborhood averaging,
- Median filtering,
- Low pass filtering, or
- Averaging of multiple images.

The details of how to implement any of the previous algorithms are beyond the scope of this paper. However we can discuss an alternative method as a conclusion of our discussion to this section. This method can be implemented with minimum computer intelligence and it does not involve complicated algorithms. First, the required edges can be determined by a semi-computerized algorithm; for example the user of the computer can use a pointing device to define similar areas between the two images (e.g., ground in both, or a tree split across the two images, or sky in both, or certain vegetation common to neighboring images). Second, the similar regions should be processed pairwise until they are close in appearance to each other. This can be done by copying the required part of the image to another location, processing it there, and then copying it back. The same thing should be done for the other pairs of specified parts until all are done. A data file could store the definition of
CHAPTER 4. CROPPING TOOL

the pairs and the processing required for each. This method might work for images which can be divided into obvious regions, but for images where there are many small regions this algorithm might be complex and unsuccessful.
Chapter 5

CONCLUSION AND FUTURE WORK

The cropping tool provides the means to manipulate images to be used with the orientation, position finding, and orienteering module of the GeoSim system. The tool can be used to crop images to provide a seamless panoramic view at any location. The differences in color and intensity between images can be minimized using the current tool but they cannot be eliminated. Storing the parameters that result from using the tool will allow the user to reverse actions and retrieve the original image at any later time. The tool will give the user fast, accurate cropping results and fairly good image enhancement results.

The user interface to the tool is designed to be efficient and easy to use. Interaction with the tool is accomplished through a minimum number of keys. Several users have all felt very comfortable with the interface.

More work is needed to insure that the output of the cropping tool is usable by the orienteering module. The tool should be used with all the images in the database, and this means that the rest of the images must be captured and processed.

Using the tool with MD/DOS IP [5], which is a communication software package that allows for mounting systems over the network as drives, might be useful in the sense that the images could be captured, processed, stored, and accessed through the network. The clients can mount the server as a normal disk drive and run the tool which will access the images from the server. This configuration will save some money because the clients will not need high capacity hard drives.
REFERENCES


Appendix A

Program Flowcharts
Figure A.1: Flow chart for the First Screen.
Figure A.2: Flow chart for the Second Screen.
Appendix B

Data Flow Diagram
Figure B.1: The main components of the tool.
Appendix C
Source Code Modules
Table C.1: The source code modules for the cropping tool.

<table>
<thead>
<tr>
<th>Module Name</th>
<th>Module Size (number of lines of code)</th>
<th>Full Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cropim.c</td>
<td>922</td>
<td>crop image</td>
<td>Crop the image on screen, maintain the data structure, and change the color and intensity of the image.</td>
</tr>
<tr>
<td>choosei.c</td>
<td>651</td>
<td>choose image</td>
<td>Display and help user select images for first screen.</td>
</tr>
<tr>
<td>crtool.c</td>
<td>223</td>
<td>crop tool</td>
<td>Initialize screen and VRAM.</td>
</tr>
<tr>
<td>imload.c</td>
<td>48</td>
<td>image load</td>
<td>Decompress and load images.</td>
</tr>
<tr>
<td>misc.c</td>
<td>399</td>
<td>miscellaneous</td>
<td>Process error messages, menus, and user input.</td>
</tr>
</tbody>
</table>